



US Patent & Trademark Office

[Subscribe \(Full Service\)](#) [Register \(Limited Service, Free\)](#) [Login](#)Search: ☐ The ACM Digital Library ☒ The Guide**THE GUIDE TO COMPUTING LITERATURE**[Feedback](#) [Report a problem](#) [Satisfaction survey](#)**Automatic inference of models for statistical code compression****Full text** [Pdf \(631 KB\)](#)

Source [Conference on Programming Language Design and Implementation](#) [archive](#)
[Proceedings of the ACM SIGPLAN 1999 conference on Programming language design and implementation](#) [table of contents](#)
Atlanta, Georgia, United States
Pages: 242 - 246
Year of Publication: 1999
ISSN:0362-1340
[Also published in ...](#)

Author [Christopher W. Fraser](#) Microsoft Research, One Microsoft Way, Redmond, WA**Sponsors** [SIGSOFT](#): ACM Special Interest Group on Software Engineering
[SIGPLAN](#): ACM Special Interest Group on Programming Languages**Publisher** ACM Press New York, NY, USA**Additional Information:** [abstract](#) [references](#) [citations](#) [index terms](#) [collaborative colleagues](#)**Tools and Actions:** [Discussions](#) [Find similar Articles](#) [Review this Article](#)
[Save this Article to a Binder](#) [Display in BibTex Format](#)**DOI Bookmark:** Use this link to bookmark this Article: <http://doi.acm.org/10.1145/301618.301672>
[What is a DOI?](#)↑ **ABSTRACT**

This paper describes experiments that apply machine learning to compress computer programs, formalizing and automating decisions about instruction encoding that have traditionally been made by humans in a more ad hoc manner. A program accepts a large training set of program material in a conventional compiler intermediate representation (IR) and automatically infers a decision tree that separates IR code into streams that compress much better than the undifferentiated whole. Driving a conventional arithmetic compressor with this model yields code 30% smaller than the previous record for IR code compression, and 24% smaller than an ambitious optimizing compiler feeding an ambitious general-purpose data compressor.

↑ **REFERENCES**

Note: OCR errors may be found in this Reference List extracted from the full text article. ACM has opted to expose the complete List rather than only correct and linked references.

1 [Timothy C. Bell](#), [John G. Cleary](#), [Ian H. Witten](#), [Text compression](#), Prentice-Hall, Inc., Upper Saddle River, NJ, 1990

2 M. Burrows and D. J. Wheeler. A block-sorting lossless data compression algorithm. Digital SRC research report 124, 5/10/94.



US Patent & Trademark Office

[Subscribe \(Full Service\)](#) [Register \(Limited Service, Free\)](#) [Login](#)

 Search: ☐ The ACM Digital Library ☒ The Guide

THE GUIDE TO COMPUTING LITERATURE


[Feedback](#) [Report a problem](#) [Satisfaction](#)
[survey](#)
[Conference on Programming Language Design and Implementation](#) [archive](#)

 Proceedings of the ACM SIGPLAN 2000 conference on Programming language design and implementation [citation](#)

 2000, Vancouver, British Columbia, Canada *June 18 - 21, 2000*
Table of Contents
Dynamo: a transparent dynamic optimization system

Vasanth Bala, Evelyn Duesterwald, Sanjeev Banerjia


Pages: 1 - 12

 Full text available:  Pdf(156 KB)

 Additional Information: [full citation](#), [abstract](#), [references](#), [citings](#), [index terms](#)
Practicing JUDO: Java under dynamic optimizations

Michał Cierniak, Guei-Yuan Lueh, James M. Stichnoth


Pages: 13 - 26

 Full text available:  Pdf(190 KB)

 Additional Information: [full citation](#), [abstract](#), [references](#), [citings](#), [index terms](#)
Split-stream dictionary program compression

Steven Lucco

Pages: 27 - 34

 Full text available:  Pdf(90 KB)

 Additional Information: [full citation](#), [abstract](#), [references](#), [citings](#), [index terms](#)
Unification-based pointer analysis with directional assignments

Manuvir Das


Pages: 35 - 46

 Full text available:  Pdf(553 KB)

 Additional Information: [full citation](#), [abstract](#), [references](#), [citings](#), [index terms](#)
Off-line variable substitution for scaling points-to analysis

Atanas Rountev, Satish Chandra

Pages: 47 - 56

 Full text available:  Pdf(220 KB)

 Additional Information: [full citation](#), [abstract](#), [references](#), [citings](#), [index terms](#)
Modular interprocedural pointer analysis using access paths: design, implementation, and evaluation

Ben-Chung Cheng, Wen-Mei W. Hwu

Pages: 57 - 69

 Full text available:  Pdf(855 KB)

 Additional Information: [full citation](#), [abstract](#), [references](#), [citings](#), [index terms](#)
Safety checking of machine code

Zhichen Xu, Barton P. Miller, Thomas Reps

Pages: 70 - 82

 Full text available:  Pdf(307 KB)

 Additional Information: [full citation](#), [abstract](#), [references](#), [citings](#), [index terms](#)
Translation validation for an optimizing compiler

George C. Necula

Pages: 83 - 94

Full text available:  Pdf(679 KB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citings](#), [index terms](#)

A certifying compiler for Java

Christopher Colby, Peter Lee, George C. Necula, Fred Blau, Mark Plesko, Kenneth Cline

Pages: 95 - 107

Full text available:  Pdf(792 KB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citings](#), [index terms](#)

Bidwidth analysis with application to silicon compilation

Mark Stephenson, Jonathan Babb, Saman Amarasinghe

Pages: 108 - 120

Full text available:  Pdf(931 KB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citings](#), [index terms](#)

Optimal instruction scheduling using integer programming

Kent Wilken, Jack Liu, Mark Heffernan

Pages: 121 - 133


Full text available:  Pdf(831 KB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citings](#), [index terms](#)

Improved spill code generation for software pipelined loops

Javier Zalamea, Josep Llosa, Eduard Ayguadé, Mateo Valero

Pages: 134 - 144

Full text available:  Pdf(688 KB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citings](#), [index terms](#)

Exploiting superword level parallelism with multimedia instruction sets

Samuel Larsen, Saman Amarasinghe

Pages: 145 - 156

Full text available:  Pdf(679 KB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citings](#), [index terms](#)

Compiler analysis of irregular memory accesses

Yuan Lin, David Padua

Pages: 157 - 168

Full text available:  Pdf(891 KB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citings](#), [index terms](#)

Transforming loops to recursion for multi-level memory hierarchies

Qing Yi, Vikram Adve, Ken Kennedy

Pages: 169 - 181


Full text available:  Pdf(933 KB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citings](#), [index terms](#)

Symbolic bounds analysis of pointers, array indices, and accessed memory regions

Radu Rugina, Martin Rinard

Pages: 182 - 195


Full text available:  Pdf(981 KB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citings](#), [index terms](#)

A framework for interprocedural optimization in the presence of dynamic class loading

Vugranam C. Sreedhar, Michael Burke, Jong-Deok Choi

Pages: 196 - 207


Full text available:  Pdf(576 KB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citings](#), [index terms](#)

Effective synchronization removal for Java

Erik Ruf

Pages: 208 - 218


Full text available:  Pdf(820 KB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citings](#), [index terms](#)

Type-based race detection for Java

Cormac Flanagan, Stephen N. Freund

Pages: 219 - 232


Full text available:  Pdf(237 KB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citings](#), [index terms](#)

On loops, dominators, and dominance frontier

G. Ramalingam

Pages: 233 - 241


Full text available:  Pdf(207 KB)

Additional Information: [full citation](#), [references](#), [citings](#), [index terms](#)

Functional reactive programming from first principles

Zhanyong Wan, Paul Hudak

Pages: 242 - 252


Full text available:  Pdf(699 KB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citings](#), [index terms](#)

Scalable context-sensitive flow analysis using instantiation constraints

Manuel Fähndrich, Jakob Rehof, Manuvir Das

Pages: 253 - 263

Full text available:  Pdf(734 KB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citings](#), [index terms](#)

Contaminated garbage collection

Dante J. Cannarozzi, Michael P. Plezbert, Ron K. Cytron

Pages: 264 - 273


Full text available:  Pdf(559 KB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citings](#), [index terms](#)

A generational on-the-fly garbage collector for Java

Tamar Domani, Elliot K. Kolodner, Erez Petrank

Pages: 274 - 284


Full text available:  Pdf(564 KB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citings](#), [index terms](#)

A single intermediate language that supports multiple implementations of exceptions

Norman Ramsey, Simon Peyton Jones

Pages: 285 - 298


Full text available:  Pdf(901 KB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citings](#), [index terms](#)

Efficient algorithms for bidirectional debugging

Bob Boothe

Pages: 299 - 310

Full text available:  Pdf(474 KB)

Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Caching function calls using precise dependencies

Allan Heydon, Roy Levin, Yuan Yu

Pages: 311 - 320


Full text available:  Pdf(244 KB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citings](#), [index terms](#)

ABCD: eliminating array bounds checks on demand

Rastislav Bodik, Rajiv Gupta, Vivek Sarkar

Pages: 321 - 333


Full text available:  Pdf(307 KB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Field analysis: getting useful and low-cost interprocedural information

Sanjay Ghemawat, Keith H. Randall, Daniel J. Scales

Pages: 334 - 344


Full text available:  [Pdf\(687 KB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

An automatic object inlining optimization and its evaluation

Julian Dolby, Andrew Chien

Pages: 345 - 357

Full text available:  [Pdf\(877 KB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2004 ACM, Inc.

[Terms of Usage](#) [Privacy Policy](#) [Code of Ethics](#) [Contact Us](#)

Useful downloads:  [Adobe Acrobat](#)  [QuickTime](#)  [Windows Media Player](#)  [Real Player](#)

IEEE HOME | SEARCH IEEE | SHOP | WEB ACCOUNT | CONTACT IEEE



Membership Publications/Services Standards Conferences Careers/Jobs

IEEE Xplore
 RELEASE 1.8

 Welcome
 United States Patent and Trademark Office

[Help](#) [FAQ](#) [Terms](#) [IEEE Peer Review](#)

Quick Links

Welcome to IEEE Xplore®

- ☐ Home
- ☐ What Can I Access?
- ☐ Log-out

Tables of Contents

- ☐ Journals & Magazines
- ☐ Conference Proceedings
- ☐ Standards

Search

- ☐ By Author
- ☐ Basic
- ☐ Advanced

Member Services

- ☐ Join IEEE
- ☐ Establish IEEE Web Account
- ☐ Access the IEEE Member Digital Library

IEEE Enterprise

- ☐ Access the IEEE Enterprise File Cabinet

Print Format

Your search matched **3** of **1071730** documents.A maximum of **500** results are displayed, **15** to a page, sorted by **Relevance Descending** order.**Refine This Search:**

You may refine your search by editing the current search expression or enter a new one in the text box.

(lucco<in>au)

Search

☐ Check to search within this result set**Results Key:****JNL** = Journal or Magazine **CNF** = Conference **STD** = Standard**1 Tarmac: a language system substrate based on mobile memory***Lucco, S.E.; Anderson, D.P.;*

Distributed Computing Systems, 1990. Proceedings., 10th International Conference on , 28 May-1 June 1990

Pages:46 - 51

[\[Abstract\]](#) [\[PDF Full-Text \(500 KB\)\]](#) **IEEE CNF**
2 Delirium: an embedding coordination language*Lucco, S.; Sharp, O.;*

Supercomputing '90. Proceedings of , 12-16 Nov. 1990

Pages:515 - 524

[\[Abstract\]](#) [\[PDF Full-Text \(772 KB\)\]](#) **IEEE CNF**
3 The Rapport multimedia conferencing system-a software overview*Ensor, J.R.; Ahuja, S.R.; Horn, D.N.; Lucco, S.E.;*

Computer Workstations, 1988., Proceedings of the 2nd IEEE Conference on , March 1988

Pages:52 - 58

[\[Abstract\]](#) [\[PDF Full-Text \(532 KB\)\]](#) **IEEE CNF**
[Home](#) | [Log-out](#) | [Journals](#) | [Conference Proceedings](#) | [Standards](#) | [Search by Author](#) | [Basic Search](#) | [Advanced Search](#) | [Join IEEE](#) | [Web Account](#) | [New this week](#) | [OPAC Linking Information](#) | [Your Feedback](#) | [Technical Support](#) | [Email Alerting](#) | [No Robots Please](#) | [Release Notes](#) | [IEEE Online Publications](#) | [Help](#) | [FAQ](#) | [Terms](#) | [Back to Top](#)

Copyright © 2004 IEEE — All rights reserved

IEEE HOME | SEARCH IEEE | SHOP | WEB ACCOUNT | CONTACT IEEE



Membership Publications/Services Standards Conferences Careers/Jobs

Welcome
United States Patent and Trademark Office

Help FAQ Terms IEEE Peer Review

Quick Links

» Se

Welcome to IEEE Xplore®

- ☐ Home
- ☐ What Can I Access?
- ☐ Log-out

Tables of Contents

- ☐ Journals & Magazines
- ☐ Conference Proceedings
- ☐ Standards

Search

- ☐ By Author
- ☐ Basic
- ☐ Advanced

Member Services

- ☐ Join IEEE
- ☐ Establish IEEE Web Account
- ☐ Access the IEEE Member Digital Library

IEEE Enterprise

- ☐ Access the IEEE Enterprise File Cabinet

Print Format

Your search matched **7** of **1071730** documents.A maximum of **500** results are displayed, **15** to a page, sorted by **Relevance Descending** order.

Refine This Search:

You may refine your search by editing the current search expression or enter a new one in the text box.

split<and> (stream<in>ti)

Search

☐ Check to search within this result set

Results Key:

JNL = Journal or Magazine **CNF** = Conference **STD** = Standard1 **Design of instruction stream buffer with trace support for X86 processors***Jih-Ching Chiu; I-Huan Huang; Chung-Ping Chung;*

Computer Design, 2000. Proceedings. 2000 International Conference on , 17-Sept. 2000

Pages:294 - 299

[\[Abstract\]](#) [\[PDF Full-Text \(464 KB\)\]](#) **IEEE CNF**2 **Universal MPEG content access using compressed-domain system stream editing techniques***Ching-Yung Lin; Tseng, B.L.; Smith, J.R.;*

Multimedia and Expo, 2002. ICME '02. Proceedings. 2002 IEEE International Conference on , Volume: 2 , 26-29 Aug. 2002

Pages:73 - 76 vol.2

[\[Abstract\]](#) [\[PDF Full-Text \(716 KB\)\]](#) **IEEE CNF**3 **Hierarchical adaptive control protocol for video streaming***Linsong Cai; Mao Liu; Wael Badawy;*

Electrical and Computer Engineering, 2002. IEEE CCECE 2002. Canadian Conference on , Volume: 2 , 12-15 May 2002

Pages:940 - 944 vol.2

[\[Abstract\]](#) [\[PDF Full-Text \(479 KB\)\]](#) **IEEE CNF**4 **QoS adaptive MPEG-2 streaming based on scalable media object framework**

Sung-Ho Ahn; Min-Gyu Kang; Doo-Hyun Kim; Hyung-Chul Kim;
Information Networking, 2001. Proceedings. 15th International Conference on
Jan.-2 Feb. 2001
Pages:683 - 688

[\[Abstract\]](#) [\[PDF Full-Text \(484 KB\)\]](#) IEEE CNF

5 Stream bubbles for steady flow visualization

Bing Zhang; Pang, A.;
Computer Graphics and Applications, 2001. Proceedings. Ninth Pacific Conference on , 16-18 Oct. 2001
Pages:169 - 177

[\[Abstract\]](#) [\[PDF Full-Text \(790 KB\)\]](#) IEEE CNF

6 Stream-temperature estimation from thermal infrared images

Kay, J.; Handcock, R.N.; Gillespie, A.; Konrad, C.; Burges, S.; Naveh, N.; Boyd, D.;
Geoscience and Remote Sensing Symposium, 2001. IGARSS '01. IEEE 2001 International , Volume: 1 , 9-13 July 2001
Pages:112 - 114 vol.1

[\[Abstract\]](#) [\[PDF Full-Text \(533 KB\)\]](#) IEEE CNF

7 Scene and content analysis from multiple video streams

Guler, S.;
Applied Imagery Pattern Recognition Workshop, AIPR 2001 30th , 10-12 Oct.
Pages:119 - 125

[\[Abstract\]](#) [\[PDF Full-Text \(724 KB\)\]](#) IEEE CNF

[Home](#) | [Log-out](#) | [Journals](#) | [Conference Proceedings](#) | [Standards](#) | [Search by Author](#) | [Basic Search](#) | [Advanced Search](#) | [Join IEEE](#) | [Web Account](#) |
[New this week](#) | [OPAC Linking Information](#) | [Your Feedback](#) | [Technical Support](#) | [Email Alerting](#) | [No Robots Please](#) | [Release Notes](#) | [IEEE Online Publications](#) | [Help](#) | [FAQ](#) | [Terms](#) | [Back to Top](#)

Copyright © 2004 IEEE — All rights reserved



US Patent & Trademark Office

[Subscribe \(Full Service\)](#) [Register \(Limited Service, Free\)](#) [Login](#)

 Search: ☒ The ACM Digital Library ☐ The Guide

split-stream

SEARCH


[Feedback](#) [Report a problem](#) [Satisfaction survey](#)
Terms used **split stream**

Found 21 of 142,346

Sort results by

relevance

Display results

expanded form

[Save results to a Binder](#)[Search Tips](#)☐ Open results in a new window[Try an Advanced Search](#)[Try this search in The ACM Guide](#)

Results 1 - 20 of 21

Result page: 1 2

Relevance scale ☐ ☐ ☐ ☐ ☐**1** [Split-stream dictionary program compression](#)

Steven Lucco

 May 2000 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 2000 conference on Programming language design and implementation**, Volume 35 Issue 5
Full text available: [pdf\(89.99 KB\)](#)
 Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

This paper describes split-stream dictionary (SSD) compression, a new technique for transforming programs into a compact, interpretable form. We define a compressed program as interpretable when it can be decompressed at basic-block granularity with reasonable efficiency. The granularity requirement enables interpreters or just-in-time (JIT) translators to decompress basic blocks incrementally during program execution. Our previous approach to interpretable compression, the Byte-coded RISC ...

Keywords: compression, runtime system, virtual machine**2** [Designing a trace format for heap allocation events](#)

Trishul Chilimbi, Richard Jones, Benjamin Zorn

 October 2000 **ACM SIGPLAN Notices , Proceedings of the second international symposium on Memory management**, Volume 36 Issue 1
Full text available: [pdf\(1.53 MB\)](#)
 Additional Information: [full citation](#), [abstract](#), [citations](#), [index terms](#)

Dynamic storage allocation continues to play an important role in the performance and correctness of systems ranging from user productivity software to high-performance servers. While algorithms for dynamic storage allocation have been studied for decades, much of the literature is based on measuring the performance of benchmark programs unrepresentative of many important allocation-intensive workloads. Furthermore, to date no standard has emerged or been proposed for publishing and exchanging ...

3 [Code optimization II: Code optimization for code compression](#)

Milenko Drinić, Darko Kirovski, Hoi Vo

 March 2003 **Proceedings of the international symposium on Code generation and optimization: feedback-directed and runtime optimization**
Full text available: [pdf\(1.07 MB\)](#)
 Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)
[Publisher Site](#)


With the emergence of software delivery platforms such as Microsoft's .NET, reduced size of

transmitted binaries has become a very important system parameter strongly affecting system performance. In this paper, we present two novel pre-processing steps for code compression that explore program binaries' syntax and semantics to achieve superior compression ratios. The first preprocessing step involves heuristic partitioning of a program binary into streams with high auto-correlation. The second ...

4 Overlay & peer-to-peer networks: SplitStream: high-bandwidth multicast in cooperative environments

Miguel Castro, Peter Druschel, Anne-Marie Kermarrec, Animesh Nandi, Antony Rowstron, Atul Singh

October 2003 **Proceedings of the nineteenth ACM symposium on Operating systems principles**

Full text available:  [pdf\(847.74 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)


In tree-based multicast systems, a relatively small number of interior nodes carry the load of forwarding multicast messages. This works well when the interior nodes are highly-available, dedicated infrastructure routers but it poses a problem for application-level multicast in peer-to-peer systems. SplitStream addresses this problem by striping the content across a forest of interior-node-disjoint multicast trees that distributes the forwarding load among all participating peers. For example, i ...

Keywords: application-level multicast, content distribution, end-system multicast, peer-to-peer, video streaming

5 Bytecode compression via profiled grammar rewriting

William S. Evans, Christopher W. Fraser

May 2001 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 2001 conference on Programming language design and implementation**, Volume 36 Issue 5

Full text available:  [pdf\(1.03 MB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

This paper describes the design and implementation of a method for producing compact, bytecoded instruction sets and interpreters for them. It accepts a grammar for programs written using a simple bytecoded stack-based instruction set, as well as a training set of sample programs. The system transforms the grammar, creating an expanded grammar that represents the same language as the original grammar, but permits a shorter derivation of the sample programs and others like them. A program's de ...

Keywords: bytecode interpretation, context-free grammars, program compression, variable-to-fixed length codes

6 Software techniques for program compaction: Cold code decompression at runtime

Saumya Debray, William S. Evans

August 2003 **Communications of the ACM**, Volume 46 Issue 8

Full text available:  [pdf\(111.99 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#), [review](#),  [html\(29.19 KB\)](#)

Using a software-based technique to dynamically decompress selected code fragments during program execution.

7 Generation of fast interpreters for Huffman compressed bytecode

Mario Latendresse, Marc Feeley

June 2003 **Proceedings of the 2003 workshop on Interpreters, Virtual Machines and Emulators**

Full text available:  pdf(323.22 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)


Embedded systems often have severe memory constraints requiring careful encoding of programs. For example, smart cards have on the order of 1K of RAM, 16K of non-volatile memory, and 24K of ROM. A virtual machine can be an effective approach to obtain compact programs but instructions are commonly encoded using one byte for the opcode and multiple bytes for the operands, which can be wasteful and thus limit the size of programs runnable on embedded systems. Our approach uses canonical Huffman co ...

Keywords: Java, canonical Huffman code, code compression, decoder

8 [Profile-guided code compression](#)

Saumya Debray, William Evans

May 2002 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 2002 Conference on Programming language design and implementation**, Volume 37 Issue 5

Full text available:  pdf(178.02 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)


As computers are increasingly used in contexts where the amount of available memory is limited, it becomes important to devise techniques that reduce the memory footprint of application programs while leaving them in an executable form. This paper describes an approach to applying data compression techniques to reduce the size of infrequently executed portions of a program. The compressed code is decompressed dynamically (via software) if needed, prior to execution. The use of data compression t ...

Keywords: code compaction, code compression, code size reduction, dynamic decompression

9 [Building a scaleable geo-spatial DBMS: technology, implementation, and evaluation](#)

Jignesh Patel, JieBing Yu, Navin Kabra, Kristin Tufte, Biswadeep Nag, Josef Burger, Nancy Hall, Karthikeyan Ramasamy, Roger Lueder, Curt Ellmann, Jim Kupsch, Shelly Guo, Johan Larson, David De Witt, Jeffrey Naughton

June 1997 **ACM SIGMOD Record , Proceedings of the 1997 ACM SIGMOD international conference on Management of data**, Volume 26 Issue 2

Full text available:  pdf(1.58 MB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

This paper presents a number of new techniques for parallelizing geo-spatial database systems and discusses their implementation in the Paradise object-relational database system. The effectiveness of these techniques is demonstrated using a variety of complex geo-spatial queries over a 120 GB global geo-spatial data set.

10 [How strong is weak mutation?](#)

A. Jefferson Offutt, Stephen D. Lee

October 1991 **Proceedings of the symposium on Testing, analysis, and verification**

Full text available:  pdf(1.25 MB) Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)

11 [A code-splitting algorithm](#)

Wayne Amsbury


December 1983 **ACM SIGARCH Computer Architecture News**, Volume 11 Issue 5

Full text available:  pdf(350.44 KB) Additional Information: [full citation](#), [references](#)

12 Can networks make an organization?

Tamar Bermann, Kari Thoresen

January 1988 **Proceedings of the 1988 ACM conference on Computer-supported cooperative work**


Full text available:  [pdf\(891.20 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Based on earlier work in Scandinavia, systems development is seen as both reflecting and shaping organizations. An ongoing action-research project at a large municipal hospital in Norway is taken as an example. This is a cooperative project in which nursing supervisors and researchers participate in shaping a learning process and designing a computer system. Some inherent contradictions, conflicts and challenges are indicated. Strengthening cooperation among themselves ...

13 Algorithm 806: SPRNG: a scalable library for pseudorandom number generation

Michael Mascagni, Ashok Srinivasan

September 2000 **ACM Transactions on Mathematical Software (TOMS)**, Volume 26 Issue 3

Full text available:  [pdf\(158.69 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)


In this article we present background, rationale, and a description of the Scalable Parallel Random Number Generators (SPRNG) library. We begin by presenting some methods for parallel pseudorandom number generation. We will focus on methods based on parameterization, meaning that we will not consider splitting methods such as the leap-frog or blocking methods. We describe, in detail, parameterized versions of the following pseudorandom number generators: (i) linear congruential generators, ...

Keywords: lagged-Fibonacci generator, linear congruential generator, parallel random-number generators, random-number software, random-number tests

14 NiagaraCQ: a scalable continuous query system for Internet databases

Jianjun Chen, David J. DeWitt, Feng Tian, Yuan Wang

May 2000 **ACM SIGMOD Record , Proceedings of the 2000 ACM SIGMOD international conference on Management of data**, Volume 29 Issue 2

Full text available:  [pdf\(165.02 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)


Continuous queries are persistent queries that allow users to receive new results when they become available. While continuous query systems can transform a passive web into an active environment, they need to be able to support millions of queries due to the scale of the Internet. No existing systems have achieved this level of scalability. NiagaraCQ addresses this problem by grouping continuous queries based on the observation that many web queries share similar structures. Grouped queries ...

15 Digest of proceedings seventh IEEE workshop on hot topics in operating systems

March 29-30 1999, Rio Rico, AZ

M. Satyanarayanan

October 1999 **ACM SIGOPS Operating Systems Review**, Volume 33 Issue 4

Full text available:  [pdf\(1.67 MB\)](#) Additional Information: [full citation](#), [abstract](#), [index terms](#)


The Seventh IEEE Workshop on Hot Topics in Operating Systems was held on March 29-30 1999 at the Rio Rico Resort & Country Club, south of Tucson, Arizona. The General Chair, Peter Druschel, and the Local Arrangements Chair, John Hartman, had gone to considerable effort to make the operation of the workshop smooth and pleasant for the participants. The secluded desert locale, the effect of brilliant sunshine and blue skies on

winter-jaded northerners, and the enthusiasm and energy of the ...

16 Quantifying loop nest locality using SPEC'95 and the perfect benchmarks

Kathryn S. McKinley, Olivier Temam

November 1999 **ACM Transactions on Computer Systems (TOCS)**, Volume 17 Issue 4

Full text available:  [pdf\(635.63 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)


This article analyzes and quantifies the locality characteristics of numerical loop nests in order to suggest future directions for architecture and software cache optimizations. Since most programs spend the majority of their time in nests, the vast majority of cache optimization techniques target loop nests. In contrast, the locality characteristics that drive these optimizations are usually collected across the entire application rather than at the nest level. Researchers have studied nu ...



17 Performance modeling and code partitioning for the DS architecture

Yinong Zhang, George B. Adams

April 1998 **ACM SIGARCH Computer Architecture News , Proceedings of the 25th annual international symposium on Computer architecture**, Volume 26 Issue 3

Full text available:  [Publisher Site](#) Additional Information: [full citation](#), [abstract](#), [citations](#), [index terms](#)


DS (Decoupled-Superscalar) is a new microarchitecture that combines decoupled and superscalar techniques to exploit instruction level parallelism. Issue bandwidth is increased while circuit complexity growth is controlled with little negative impact on performance. Programs for DS are compiled into two instruction substreams: the dominant substream navigates the control flow and the rest of computational task is shared between the dominant and subsidiary substreams. Each substream is processed b ...



18 Code compression

Jens Ernst, William Evans, Christopher W. Fraser, Todd A. Proebsting, Steven Lucco

May 1997 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 1997 conference on Programming language design and implementation**, Volume 32 Issue 5

Full text available:  [pdf\(1.11 MB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)


Current research in compiler optimization counts mainly CPU time and perhaps the first cache level or two. This view has been important but is becoming myopic, at least from a system-wide viewpoint, as the ratio of network and disk speeds to CPU speeds grows exponentially. For example, we have seen the CPU idle for most of the time during paging, so compressing pages can increase total performance even though the CPU must decompress or interpret the page contents. Another profile shows that many ...



19 A quantitative analysis of loop nest locality

Kathryn S. McKinley, Olivier Temam

September 1996 **Proceedings of the seventh international conference on Architectural support for programming languages and operating systems**, Volume 31 , 30 Issue 9 , 5

Full text available:  [pdf\(1.49 MB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)


This paper analyzes and quantifies the locality characteristics of numerical loop nests in order to suggest future directions for architecture and software cache optimizations. Since most programs spend the majority of their time in nests, the vast majority of cache optimization techniques target loop nests. In contrast, the locality characteristics that drive these optimizations are usually collected across the entire application rather than the nest level. Indeed, researchers have studied nume



20 Improved serial algorithms for mutation analysis

Stewart N. Weiss, Vladimir N. Fleyshgaker

July 1993 **ACM SIGSOFT Software Engineering Notes , Proceedings of the 1993 ACM SIGSOFT international symposium on Software testing and analysis**, Volume 18 Issue 3

Full text available:  [pdf\(988.87 KB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)


Existing serial algorithms to do mutation analysis are inefficient, and descriptions of parallel mutation systems pre-suppose that these serial algorithms are the best one can do serially. We present a universal mutation analysis data structure and new serial algorithms for both strong and weak mutation analysis that on average should perform much faster than existing ones, and can never do worse. We describe these algorithms as well as the results of our analysis of their run time complexity.


Results 1 - 20 of 21

Result page: **1** 2

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2004 ACM, Inc.

[Terms of Usage](#) [Privacy Policy](#) [Code of Ethics](#) [Contact Us](#)

Useful downloads:  [Adobe Acrobat](#)

 [QuickTime](#)

 [Windows Media Player](#)

 [Real Player](#)



[Subscribe \(Full Service\)](#) [Register \(Limited Service, Free\)](#) [Login](#)

Search: ☒ The ACM Digital Library ☐ The Guide

split-stream



[Feedback](#) [Report a problem](#) [Satisfaction survey](#)

Terms used **split stream**

Found **21** of **142,346**

Sort results by

relevance



[Save results to a Binder](#)

[Try an Advanced Search](#)

[Try this search in The ACM Guide](#)

Display results

expanded form



[Search Tips](#)

☐ Open results in a new window

Results 21 - 21 of 21

Result page: [previous](#) [1](#) [2](#)

Relevance scale ☐ ☐ ☐ ☐ ☐

21 [VPC3: a fast and effective trace-compression algorithm](#)

Martin Burtscher

June 2004 **ACM SIGMETRICS Performance Evaluation Review, Proceedings of the joint international conference on Measurement and modeling of computer systems**, Volume 32 Issue 1

Full text available: [pdf\(417.20 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Trace files are widely used in research and academia to study the behavior of programs. They are simple to process and guarantee repeatability. Unfortunately, they tend to be very large. This paper describes *vpc3*, a fundamentally new approach to compressing program traces. *Vpc3* employs value predictors to bring out and amplify patterns in the traces so that conventional compressors can compress them more effectively. In fact, our approach not only results in much higher compression ...

Keywords: predictor-based compression, trace compression, trace files

Results 21 - 21 of 21

Result page: [previous](#) [1](#) [2](#)

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2004 ACM, Inc.
[Terms of Usage](#) [Privacy Policy](#) [Code of Ethics](#) [Contact Us](#)

Useful downloads: [Adobe Acrobat](#) [QuickTime](#) [Windows Media Player](#) [Real Player](#)

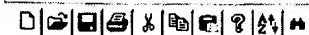


- ☐ Drafts
- ☐ Pending
- ☐ Active
- ☐ Failed
- ☐ Saved
 - ☐ (6) (382/232.ccls. or 382
 - ☐ (63) (split adj stream) s
 - ☐ (0) (split adj stream) wi
 - ☐ (0) (split\$3 adj stream)
 - ☐ (2) (split with dictionary)

 Search
DBs: USPATDefault operator: OR☐ Plurals☒ Highlight all hit terms initially

894859: split stream compression.wsp.2

	Type	Hits	Search Text	
1	BRS	6	(382/232.ccls. or 382/239.ccls. 382/240.ccls. or 382/242-243.ccls. or 382/282.ccls. or 358/537-539.ccls. or 358/1.15.ccls. or 375/240.11 or 375/240.18-240.19.ccls.) and priorit\$3 and sort\$3 and refin\$3 and significan\$2 and insignifican\$2	US
2	BRS	63	(split adj stream) same compress\$3	US
3	BRS	0	(split adj stream) with dictionary	US
4	BRS	0	(split\$3 adj stream) with dictionary	US
5	BRS	2	(split with dictionary) same compress\$3	US
6	BRS	166	ssd with generat\$3	US
7	BRS	34	(ssd with generat\$3) and compress\$3	US
8	BRS	879	split\$3 adj stream\$3	US
9	BRS	5	(base adj entr\$3) and (sequen\$3 adj entr\$3)	US
10	BRS	0	(split\$3 adj stream\$3) and ((base adj entr\$3) and (sequen\$3 adj entr\$3))	US
11	BRS	3	lucco-\$.in.	US
12	BRS	1	"4667290".PN.	US
13	BRS	1	"5386557".PN.	US
14	BRS	1	"5644709".PN.	US
15	BRS	1	"5644709".PN.	US
16	BRS	1	"5590331".PN.	US
17	BRS	1	"5590329".PN.	US
18	BRS	1	"5586323".PN.	US
19	BRS	84	fraser-c\$.in.	US
20	BRS	256	ernst-j\$.in.	US
21	BRS	458	evans-w\$.in.	US



- ☐ Drafts
- ☐ Pending
- ☐ Active
- ☐ Failed
- ☐ Saved
 - ☐ (6) (382/232.ccls. or 382
 - ☐ (63) (split adj stream) s
 - ☐ (0) (split adj stream) wi
 - ☐ (0) (split\$3 adj stream)
 - ☐ (2) (split with dictionary

Search DB: Default operator: ☐ Plurals☒ Highlight all hit terms initially

894859 split stream compression.wsp:2

	Type	Hits	Search Text	
15	BRS	1	"5644709". PN.	US
16	BRS	1	"5590331". PN.	US
17	BRS	1	"5590329". PN.	US
18	BRS	1	"5586323". PN.	US
19	BRS	84	fraser-c\$.in.	US
20	BRS	256	ernst-j\$.in.	US
21	BRS	458	evans-w\$.in.	US
22	BRS	2	proebsting-t\$.in.	US
23	BRS	0	ernst-j\$.in. and split and stream and dictionary and base and sequence	US
24	BRS	0	evans-w\$.in. and split and stream and dictionary and base and sequence	US
25	BRS	1	fraser-c\$.in. and split and stream and dictionary and base and sequence	US
26	BRS	1	fraser-c\$.in. and split and stream and dictionary	US
27	BRS	0	ernst-j\$.in. and split and stream and dictionary	US
28	BRS	0	evans-w\$.in. and split and stream and dictionary	US
29	BRS	0	ernst-j\$.in. and split and stream and base and sequence	US
30	BRS	0	evans-w\$.in. and split and stream and base and sequence	US
31	BRS	0	ernst-j\$.in. and split and stream	US
32	BRS	2	evans-w\$.in. and split and stream	US
33	BRS	1	fraser-c\$.in. and split and stream	US
34	BRS	36	717/\$.ccls. and entr\$3 and dictionary\$3 and instruction\$2 and compress\$3	US